Steven Dea
Lab #1
Basic R syntax/plots with data solutions

For this lab, we will be using some basic data manipulation and plotting commands in R. We are working with a data set that is comparing the transcript profiles from peripheral B lymphocytes between patients with systemic lupus erythematosus (SLE) and normal healthy controls. The GEO summary of the data set is as follows:

Systemic lupus erythematosus (SLE) is an autoimmune disease with an important clinical and biological heterogeneity. B lymphocytes appear central to the development of SLE which is characterized by the production of a large variety of autoantibodies and hypergammaglobulinemia. In mice, immature B cells from spontaneous lupus prone animals are able to produce autoantibodies when transferred into immunodeficient mice, strongly suggesting the existence of intrinsic B cell defects during lupus. In order to approach these defects in humans, we compared the peripheral B cell transcriptomes of quiescent lupus patients to normal B cell transcriptomes.

1.) Go to class website under Course Documents > Data Sets and download the SLE B cell data set (from Garaud et al).

2.) Unzip the text file, and read into R (Hint: using the read.table() function with a "header=T" argument and "row.names=1" argument is one method to do this).

> lab1=read.table("/Users/stevendea/Desktop/JHU/Fall 2019/Gene Expression Data Analysis and Visualization/Labs/Lab1/sle_b_cell.txt", header=TRUE, row.names=1)

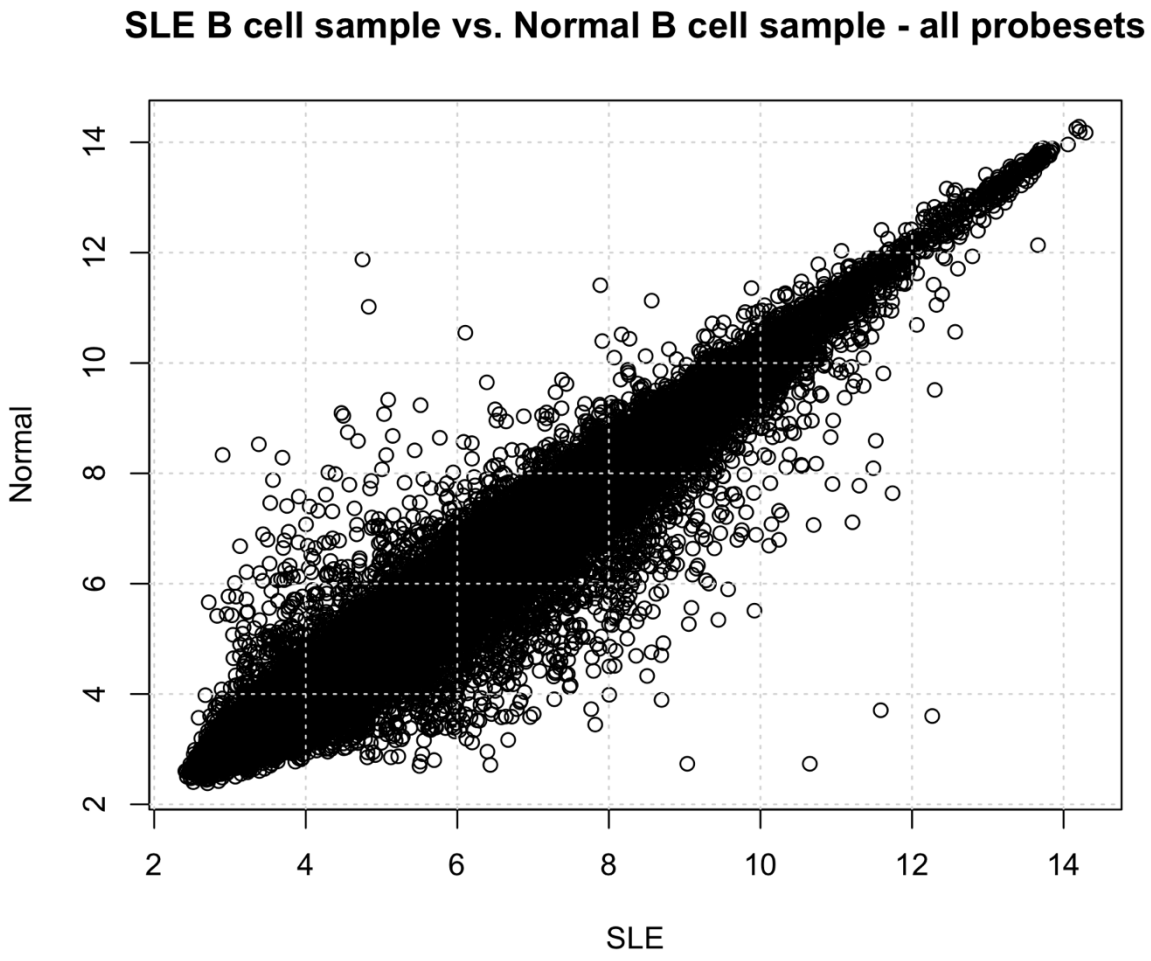
3.) Look at the dimensions of the data. There should be 26 samples. If you have 27 samples, you still have the row names in the first data column, so retry 2 to set the row names to these.

4.) Print the sample names to screen.

> lab1[0,]
[1] sle.1 sle.2 sle.3 sle.4 sle.5 sle.6 sle.7 sle.8 sle.9 sle.10 sle.11 sle.12 sle.13 sle.14 sle.15 sle.16 sle.17 control.1
[19] control.2 control.3 control.4 control.5 control.6 control.7 control.8 control.9
<0 rows> (or 0-length row.names)

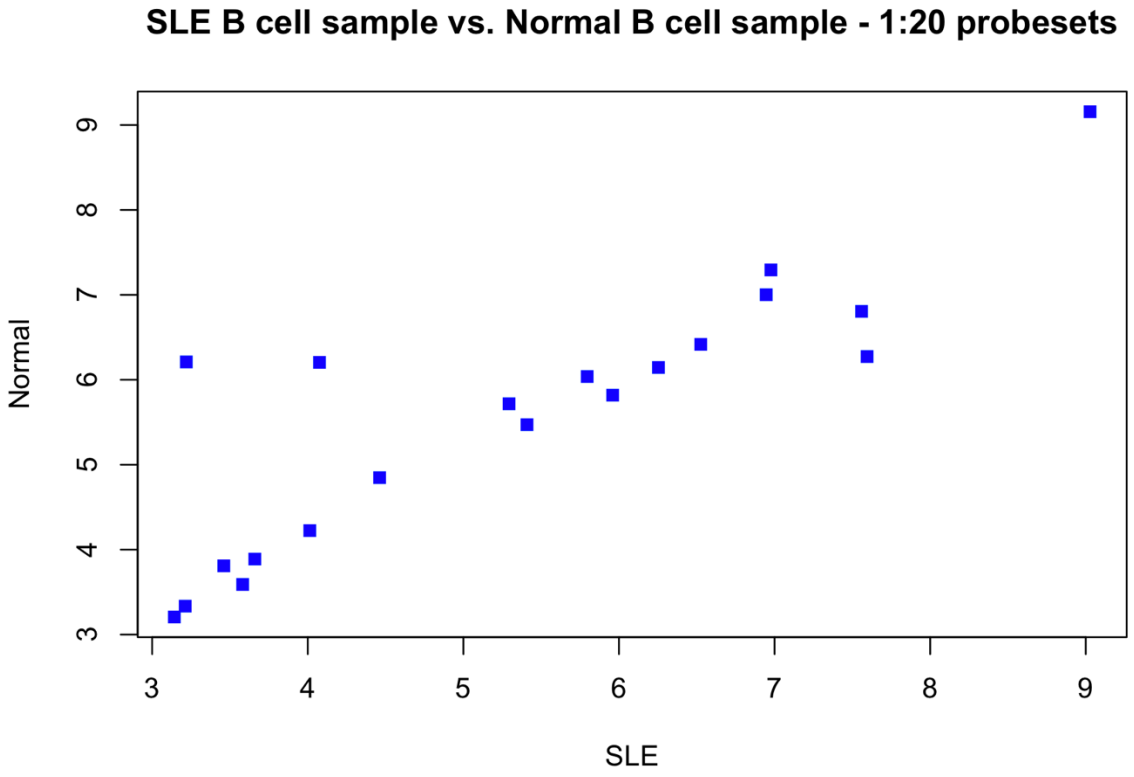
5.) Plot the second SLE patient sample versus the first normal control samples in an xy scatter plot. Remember that the first argument is the x vector. Label the x and y-axes as 'Normal' and 'SLE', respectively. Title the plot, 'SLE B cell sample vs. Normal B cell sample – all probesets'. Add grey grid lines with the function grid().

> plot(sle.2, control.1, main = "SLE B cell sample vs. Normal B cell sample - all probesets", xlab="SLE", ylab="Normal")
> grid()



6.) Now do the same plot but pick only the first 20 probesets. Use the pch=15 argument to change the shape and color the points blue with the col argument.

> plot(sle.2[1:20], control.1[1:20], main = "SLE B cell sample vs. Normal B cell sample - 1:20 probesets", xlab="SLE", ylab="Normal", pch=15, col=c("blue"))

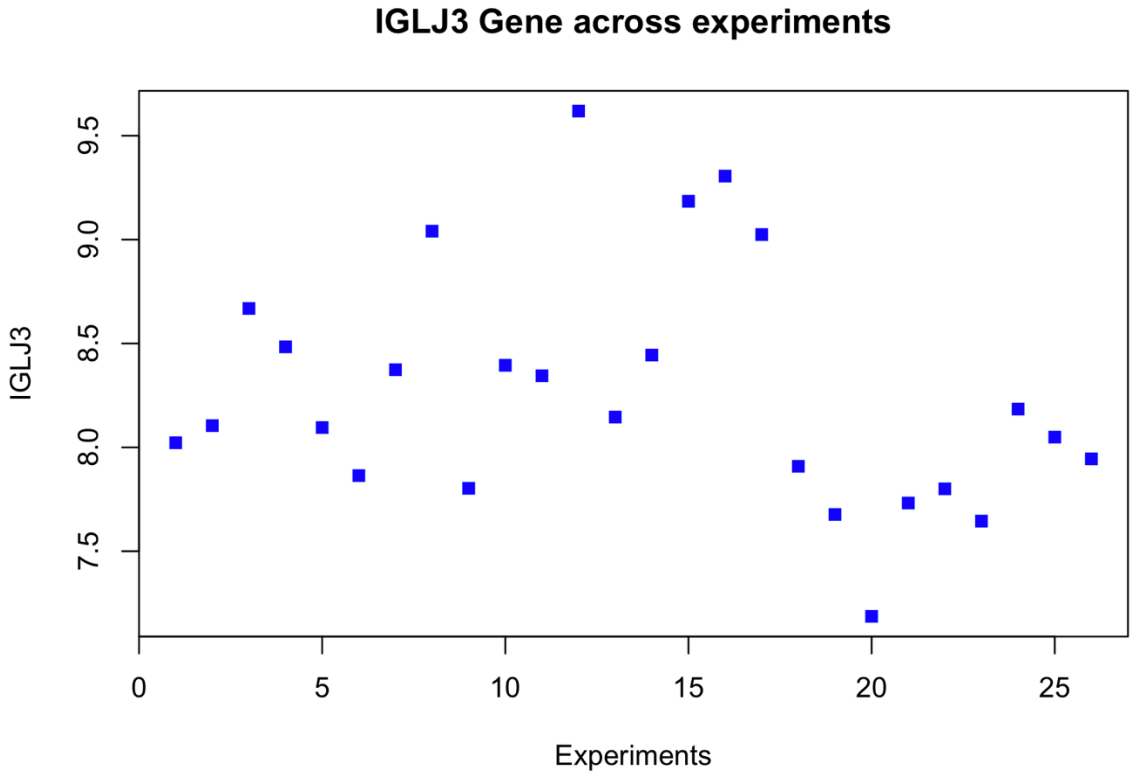


7.) Now plot the following gene in a gene profile plot, IGLJ3 (immunoglobulin lambda joining 3), which is probeset ID 211881_x_at. This type of plot has the sample indices across the x-axis and the intensities on the y-axis, so you can see a profile of the gene across experiments or arrays. First plot the ranges using the type="n" argument and the plot() function, then add the genes with the lines() function call. Add grid lines. Hint: to plot just ranges of x and y vectors, use the range() function like so:

plot(range(1:26),range(dat[geneX,]),...

Be sure to cast the gene vector to numeric before plotting.

>IGLJ3<-lab1["211881_x_at",]
>IGLJ3n<- as.numeric(IGLJ3)
> plot(IGLJ3n, main="IGLJ3 Gene across experiments", ylab="IGLJ3", xlab="Experiments", pch=15, col="blue")

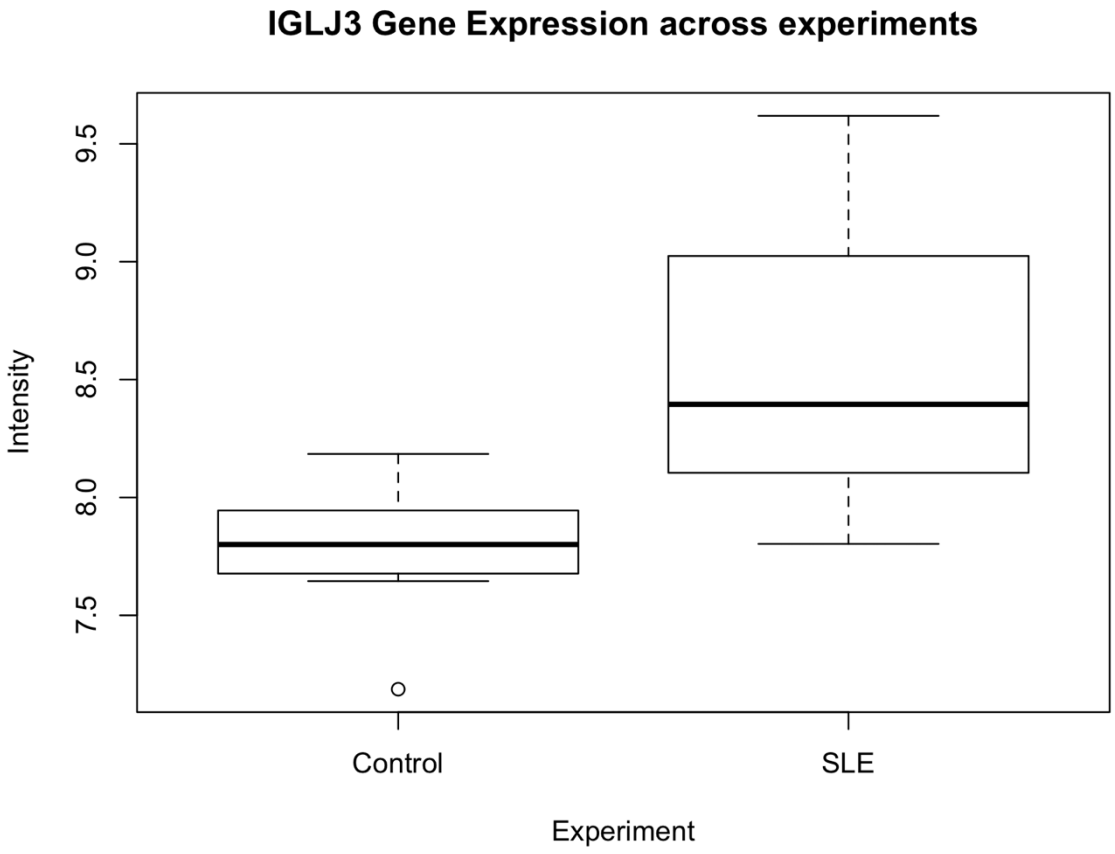


8.) Finally, another way to visualize a gene profile across conditions is to graph a boxplot with a single distribution box per condition. To do this, we need to create a factor vector that indicates the disease or normal condition like so:

f <- c(rep("SLE",17),rep("Control",9))

Then use this vector with the expression vector for IGLJ3 in the boxplot function to create the graph.

> f<-c(rep("SLE",17), rep("Control",9))
> boxplot(IGLJ3n~f, main="IGLJ3 Gene Expression across experiments", ylab="Intensity", xlab="Experiment")



Not required, but you can increase the plot info by using the with() function and stripchart() function to add points.